Qualification specification

NCFE Level 3 Certificate in Coding Practices
QN: 603/5793/9

Contents

Summary of changes

This document summarises the changes to this qualification specification since the last version (Version 1.0 July 2020). Please check the NCFE website for the most recent version.

Version	Publication date	Summary of amendments
v1.0	July 2020	First publication
v1.1	June 2022	Further information added to the how the qualification is assessed section to confirm that unless otherwise stated in this specification, all learners taking this qualification must be assessed in English and all assessment evidence presented for external quality assurance must be in English. Information added to the entry guidance section to advise that registration is at the discretion of the centre, in accordance with equality legislation and should be made on the Portal. Information added to the support handbook section about how to access support handbooks.

Section 1

About this qualification

About this qualification

This Qualification Specification contains details of all the units and assessments required to complete this qualification.

To ensure that you are using the most up-to-date version of this Qualification Specification, please check the version number and date in the page footer against that of the Qualification Specification on the NCFE website.

If you advertise this qualification using a different or shortened name, you must ensure that learners are aware that their final certificate will state the full regulated qualification title.

Support Handbook

This qualification specification must be used alongside the mandatory support handbook which can be found on the NCFE website. This contains additional supporting information to help with planning, delivery and assessment.

This qualification specification contains all the qualification-specific information you will need that is not covered in the support handbook.

Qualification summary	
Qualification title	NCFE Level 3 Certificate in Coding Practices
Qualification number (QN)	603/5793/9
Aim reference	60357939
Total Qualification Time (TQT)	260
Guided Learning Hours (GLH)	180
Minimum age	16
Qualification purpose	This qualification is designed for learners who want to increase their knowledge, skills and understanding of coding. Successful completion of the required units will allow the learner to develop in-depth knowledge, skills and understanding of coding practices. It will also support progression into relevant employment and further study in coding.
Aims and objectives	This qualification aims to: • focus on the study of coding, within the Information and Communication Technology sector • offer breadth and depth of study, incorporating a key core of knowledge • provide opportunities to acquire a number of practical and technical skills. The objectives of this qualification are to enable learners to: • understand the stages of the software development lifecycle, software methodologies and how working as a team contributes to effective software delivery • understand the purpose and use of coding standards, coding principles and practices and coding design processes • write and debug interpreted and compiled code distinguishing the different coding solutions • fix bugs using appropriate techniques and reflect on how bugs can be prevented • understand deployment, configuration management and version control systems. Throughout this qualification, we use the term ‘coding’ which is also known as ‘programming’.

Work/industry placement experience	Work/industry placement experience is not required.
Real work environment (RWE) requirement/ recommendation	Where the assessment requirements for a unit allow, it is essential that organisations wishing to operate a RWE do so in an environment which reflects a real work setting and replicates the key characteristics of the workplace in which the skill to be assessed is normally employed. This is often used to support simulation.
Rules of combination	To be awarded the Level 3 Certificate in Coding Practices, learners are required to successfully complete 5 mandatory units.
Grading	Achieved/Not Yet Achieved.
Assessment method	Internally assessed and externally quality assured portfolio of evidence.
Progression	Learners who achieve this qualification could progress to: • Level 4 Certificate in IT • Level 4 Diploma in Software Development • Software Development Technician apprenticeship
Regulation information	This is a regulated qualification. The regulated number for this qualification is 603/5793/9.
Funding	This qualification may be eligible for funding. For further guidance on funding, please contact your local funding provider.

Entry guidance

This qualification is designed for learners who are working or would like to work in coding. It is ideal for those who want to consolidate their existing knowledge and learn practical skills which can be used to assist in seeking employment in coding or to progress into further study.

Registration is at the discretion of the centre, in accordance with equality legislation, and should be made on the Portal. However, learners should be aged 16 or above to undertake this qualification.

There is no specific prior knowledge a learner must have for this qualification. However, learners may find it helpful if they've already achieved a Level 2 digital skills or information technology related qualification, and a Level 2 mathematics qualification.

Centres are responsible for ensuring that this qualification is appropriate for the age and ability of learners. They need to make sure that learners can fulfil the requirements of the learning outcomes and comply with the relevant literacy, numeracy and health and safety aspects of this qualification.

Learners registered on this qualification should not undertake another qualification at the same level with the same or a similar title, as duplication of learning may affect funding eligibility.

Achieving this qualification

To be awarded this qualification, learners are required to successfully achieve **5** mandatory units.

Please refer to the list of units below or the unit summaries in Section 2 for further information.

To achieve this qualification, learners must successfully demonstrate their achievement of all learning outcomes of the units as detailed in this Qualification Specification. A partial certificate may be requested for learners who do not achieve their full qualification but have achieved at least one whole unit.

Units

To make cross-referencing assessment and quality assurance easier, we have used a sequential numbering system in this document for each unit.

The regulated unit number is indicated in brackets for each unit (eg M/100/7116) within section 2.

 Knowledge only units are indicated by a star. If a unit is not marked with a star, it is a skills unit or contains a mix of knowledge and skills.

Mandatory units

	Unit number	Regulated unit number	Unit title	Level	GLH
	Unit 01	M/618/0976	Coding requirements and planning	3	30
	Unit 02	A/618/0978	Understand coding design	3	40
	Unit 03	F/618/0979	Implementation of coding	3	40
	Unit 04	T/618/0980	Software testing	3	40
	Unit 05	A/618/0981	Understand deployment, maintenance and configuration management	3	30

The units above may be available as stand-alone unit programmes. Please visit our website for further information.

Progression to higher level studies

This qualification aims to provide learners with a number of progression options, including higher level studies at university or FE colleges. The skills required to progress to higher academic studies are different from those required at Levels 1 and 2. Level 3 qualifications enable the development of these skills. Although there is no single definition of higher level learning skills, they include:

- checking and testing information
- supporting points with evidence
- self-directed study
- self-motivation
- thinking for yourself
- analysing and synthesising information/materials
- critical thinking and problem solving
- working collaboratively
- reflecting upon learning and identifying improvements.

Level 3 criteria can require learners to **analyse**, **draw conclusions**, **interpret** or **justify**, which are all examples of higher level skills. This means that evidence provided for the portfolio will also demonstrate the development and use of higher level learning skills.

If you need any further information, please refer to the NCFE website.

How the qualification is assessed

Assessment is the process of measuring a learner's skill, knowledge and understanding against the standards set in a qualification.

This qualification is internally assessed and externally quality assured.

The assessment consists of one component:

- an internally assessed portfolio of evidence which is assessed by centre staff and externally quality assured by NCFE.

Learners must be successful in **this** component to gain the Level 3 Certificate in Coding Practices.

Learners who aren't successful can resubmit work within the registration period; however, a charge may apply.

All the evidence generated by the learner will be assessed against the standards expected of a Level 3 learner for each learning outcome.

Unless stated otherwise in this qualification specification, all learners taking this qualification must be assessed in English and all assessment evidence presented for external quality assurance must be in English.

Internal assessment

Each learner must create a portfolio of evidence generated from appropriate assessment tasks, which demonstrates achievement of all the learning outcomes associated with each unit. On completion of each unit, learners must declare that the work produced is their own and the Assessor must countersign this. Examples of suitable evidence for the portfolio for each unit are provided in Section 2.

Internally assessed work should be completed by the learner in accordance with the Qualification Specification.

The Tutor must be satisfied that the work produced is the learner's own.

A centre may choose to create their own internal assessment tasks. The tasks should:

- be accessible and lead to objective assessment judgements
- permit and encourage authentic activities where the learner's own work can be clearly judged
- refer to Course File Documents on the NCFE website.

Supervision of learners and your role as an Assessor

Guidance on how to administer the internal assessment and the support you provide to learners can be found on the NCFE website.

Feedback to learners

Guidance on providing feedback during teaching and learning and each stage of the assessment can be found on the NCFE website.

Section 2

Unit content and assessment guidance

Unit content and assessment guidance

This section provides details of the structure and content of this qualification.

The types of evidence listed are for guidance purposes only. Within learners' portfolios, other types of evidence are acceptable if all learning outcomes are covered and if the evidence generated can be internally and externally quality assured. For approval of methods of internal assessment other than portfolio building, please contact our Quality Assurance team.

The Explanation of terms explains how the terms used in the unit content are applied to this qualification. This document can be found in Section 3.

For further information or guidance about this qualification, please contact our Customer Support team.

Unit 01 Coding requirements and planning (M/618/0976)

Unit summary	In this unit, the learner will understand software methodologies in project management and the process of requirements capture. They will also learn about risk management and how working as a team contributes to effective software delivery.
Guided learning hours	30
Level	3
Mandatory/optional	Mandatory

Learning outcome 1

The learner will:

1 Understand software methodologies in project management

The learner can:

1.1 Summarise the key features of the following:
- scrum
- kanban
- waterfall
- agile

1.2 Describe the similarities and differences between:
- scrum
- kanban
- waterfall
- agile

1.3 Consider the strengths and weaknesses of each methodology

1.4 Give an example of when each methodology is most suitable

---

Learning outcome 2

The learner will:

2 Understand the capture and documentation of requirements

The learner can:

2.1 Identify the process of requirements capture

2.2 Describe the following terms:
- epics
- features
- user stories
- backlogs

2.3 Explain alternative methods of documenting requirements

Learning outcome 3

The learner will:

3 Understand risk management

The learner can:

3.1 Explain what is meant by risk management
3.2 Describe potential risks that could affect project delivery
3.3 Describe how to reduce risk and follow risk mitigation
3.4 Explain what is meant by a project's critical path

Learning outcome 4

The learner will:

4 Understand how teamwork contributes to effective delivery of software

The learner can:

4.1 Explain the need to develop working relationships with other teams
4.2 Identify the benefits of developing positive working relationships with other teams and departments
4.3 Explain conflict management techniques that may be used for effective team collaboration
4.4 Explain the potential impact of ineffective working relationships with other teams and departments
4.5 Explain how team members can reflect on progress and identify improvements

---

Assessment guidance

Delivery and assessment
1.1 Learners will need to summarise key features considering the following: • the key aims of the methodology • any dependencies/requirements for use • roles and responsibilities • how work is divided between team members • how the Software Delivery Lifecycle is utilised • tools and processes • checkpoints and milestones • how progress is measured • any impact to the customer. 1.2 The learner should be able to apply their knowledge to describe similarities and differences between a given set of methodologies. The points used in the previous learning outcome can be used to support this. 1.3 The learner will need to consider the strengths and weaknesses of each methodology and should be able to consider, with sound reasoning, the potential strengths and weaknesses of following them. The learner should consider: • requirements collection • task breakdown • task workflow • change management • the feedback loop. 1.4 The learner will be able to give examples of different methodologies to use and the circumstances when they become more suitable/ unsuitable for selection. Choose under what conditions different methodologies become more suitable/unsuitable for selection. The learner should consider: • project and product types • requirements and documentation • compliance and governanc • visibility and scope • plan and timescales • staffing and resourcing • volatility. Learners could be given a scenario where they have to select the appropriate methodology to suit that particular task. 2.1 The learner will gain an understanding that requirements can come from many sources, and that there are various techniques used for extracting them and tools and processes used for documenting and enhancing them. The learner must include: • different types of requirements (business, functional, non-functional etc) • roles and responsibilities during requirements capture

- backlog processes such as grooming and refinement.

2.2 The learner must describe the structure of a software development backlog and have the ability to provide a concise description of the following terms:

- backlog
- epics
- features
- stories.

2.3 The learner should be able to respond correctly to a set of given requirements with the knowledge that often requirements are not always complete or hold all required information.

3.3 The learner will need to describe how to reduce the risk and follow mitigation, taking into consideration the following:

- missing/unclear requirements
- skills shortage
- time
- budget
- resource shortage
- decision-making.

For each of the identified risk types, the learner should be able to propose potential mitigations. The learner will need to consider key elements of the following:

- scrum
- kanban
- waterfall
- agile.

The Explanation of terms (Section 3) explains how the terms used in the unit content are applied to this qualification.

Types of evidence

Evidence could include:

- research
- reports
- learner written statement
- written or oral questions and answers
- discussion
- assignment
- presentation.

Unit 02 Understand coding design (A/618/0978)

Unit summary	In this unit, the learner will understand the purpose and use of coding standards. They will also learn about good coding principles and practices, and this will help them to understand the coding design process.
Guided learning hours	40
Level	3
Mandatory/optional	Mandatory

Learning outcome 1

The learner will:

1 Understand the use of coding standards

The learner can:

1.1 Identify the importance of coding standards
1.2 Explain the consequences of not following coding standards
1.3 Explain the processes and tools that could be used to promote coding standards

---

Learning outcome 2

The learner will:

2 Understand the purpose of good coding principles and practices

The learner can:

2.1 Identify good coding principles and practices used by Software Developers
2.2 Explain why it is important to follow good coding principles and practices
2.3 Draw conclusions from the use of **coding practices and principles**

Learning outcome 3

The learner will:

3 Know about coding reviews

The learner can:

3.1 Identify different types of coding review
3.2 Explain why code is reviewed
3.3 Explain why it is important to undertake a coding review
3.4 Explain the importance of effective feedback in a coding review
3.5 Distinguish the difference between submitting and performing a coding review
3.6 Explain the advantages and disadvantages of:
- pair programming
- mob programming

Learning outcome 4

The learner will:

4 Understand databases

The learner can:

4.1 Explain what is meant by:
- structured data
- unstructured data

4.2 Describe what is meant by databases
4.3 Describe the structure and key components of a database table
4.4 Explore the use of primary and foreign keys
4.5 Describe techniques for querying databases
4.6 Explain the purpose of database indexing

Learning outcome 5

The learner will:

5 Understand the principles of interface design

The learner can:

5.1 Identify the elements of UX (User Experience) design
5.2 Describe the differences between UX and UI (User Interface)
5.3 Explain the key skills required for creative problem-solving
5.4 Describe the key methods and techniques used in creative problem-solving
5.5 Explain the importance of information architecture and UX
5.6 Explain the principles of good user interface design

Learning outcome 6

The learner will:

6 Understand the importance of building security by design into software at the development stage

The learner can:

6.1 Describe what is meant by building security by design into software at the development stage
6.2 Explain methods used to ensure software is secure
6.3 Identify types of security issues and threats that can impact software development

---

Key word(s)

Coding practices and principles: the basic common principles, including DRY (don't repeat yourself), defensive programming, commenting, refactoring, patterns/anti-pattern.

Assessment guidance

Delivery and assessment
1.2 The learner must be able to explain the potential implications of following their own methods and how doing this could cause adverse effects to the software in areas such as security and performance by working in untried and untested ways. 1.3 The learner will be able to explain processes and tools that can assist a Development team in ensuring coding standards are followed, considering manual methods such as code reviews and static document analysis. There should also be an understanding of technical tooling that can be applied and run manually, on save or be integrated into the check-in and build processes. 2.1 The learner should be able to identify sets of coding principles used widely in the industry today such as KISS (keep it simple stupid), or SOLID (single responsibility principle, open-closed principle, Liskov substitution principle, interface segregation principle, and dependency inversion principle), and know and understand the principles within them. 2.2 The learner must be able to explain the importance of following good coding principles, the impact this has on the code that they write and how it can benefit themselves and their peers. 2.3 The learner should consider known sets of coding practices and principles and draw conclusions on how the adherence to these guidelines would change the code they write, as well as the positive and negative implications of having to follow coding practices and principles, versus those of not following. 3.1 The learner must identify different types of review that can be used within Software Development teams, the interactions that take place in each and the impact they can have on quality and the productivity of a Software Development team. The learner should be able to note when and why the different types of review might take place. 3.3 The learner must be able to explain clearly the reasons why code review is an important step in the development process, the things that can be ‘caught’ during a code review, the issues that could arise if code is not reviewed and the different reasons a code review might be requested or performed. 3.5 The learner should know the difference between the 2 tasks of submitting and performing a coding review, the importance of both giving and receiving feedback and the actions that take place during and after code review activities. The learner should understand why it is important to review the code and not the Developer. 3.6 The learner will understand what these programming techniques are, how they can be integrated into the development process and the advantages and disadvantages brought by such activities. 4.1 The learner must understand what is meant by the term ‘structured data’ (what type of data is classed as ‘structured’) and be able to demonstrate a good understanding of how structured data is organised. Similarly, the learner should understand types of ‘unstructured data’, how it is stored and why it is different. 4.2 The learner will describe the meaning of the word ‘database’, the different types of databases and the data they can store. The learner should understand the difference between NoSQL and relational databases, and they should have an understanding of the key technologies in use today and know that data can be stored both online (cloud-based) or on premises.

4.3 The learner will understand the structure of a database table and be familiar with key terms such as ‘record’, ‘field’ and ‘key’.

4.4 The learner will explore the use of primary and foreign keys, understanding the differences between them, their relationship to one another, the qualities they possess and their purpose. The learner should understand the concepts of constraints and relational integrity, why primary keys must be unique and what the term ‘auto-increment’ means.

4.5 The learner will be familiar with query language and the concept of querying a database. They should be able to describe the different ways (and places from) a query can be executed. The learner will understand basic CRUD (create, read, update and delete) operations and stored procedures.

5.1 The learner will be able to identify elements of UX design and why they are important whilst designing and building user interfaces.

5.2 The learner should know the meaning of both terms (UX and UI) and be familiar with the core concepts within each. The learner should be able to communicate the difference between UI and UX and why the 2 are not the same.

5.3 The learner should explain the key skills of creative problem-solving. The learner should be aware of the role that creativity plays in creating good user experiences.

5.4 The learner should be able to describe tools and processes that may help during the creative process and with testing new ideas.

5.5 The learner should be able to explain the meaning of HCI (human–computer interaction) and why it is important. The learner should be able to provide examples of good and bad AI (artificial intelligence) and give sound reasoning as to why.

5.6 The learner should explain the principles of interface design and how a badly designed interface can affect a product, in ways such as:

- increasing the bounce rate of a website
- posing risks to data integrity
- affecting the performance of a website.

6.1 The learner should describe what is meant by security by design and how it differs to the standard SDLC (Software Development Life Cycle).The learner should show an understanding of why security should be considered as early as possible and the potential impacts of not doing so.

6.2 The learner needs to explain methods that can be used to secure software. Key steps may include:

- data encoding
- input validation
- the use of HTTPS (Hyper Text Transport Protocol Secure)
- adequate password policy
- handling errors correctly.

6.3 The learner must be able to identify some of the most common security issues and threats. These may include:

- injection
- denial of service
- malware
- phishing
- social engineering.

The Explanation of terms (Section 3) explains how the terms used in the unit content are applied to this qualification.

Types of evidence

Evidence could include:

- research
- reports
- learner written statement
- written or oral questions and answers
- discussion
- assignment
- presentation.

Unit 03 Implementation of coding (F/618/0979)

Unit summary	In this unit, the learner will understand tools and key concepts used in coding. They will also be able to write and debug interpreted and compiled code distinguishing the different coding solutions.
Guided learning hours	40
Level	3
Mandatory/optional	Mandatory

Learning outcome 1

The learner will:

1 Understand tools used in coding

The learner can:

1.1 Identify tools used in coding
1.2 Select the appropriate development environment for different **languages**
1.3 Describe what is meant by source code control
1.4 Evaluate source code control solutions

Learning outcome 2

The learner will:

2 Understand key concepts of coding

The learner can:

2.1 Explain these common coding concepts:
- iterations
- sequences
- selection
- data
- methods

2.2 Explain these object orientated programming concepts:
- objects
- fields
- interfaces
- classes
- properties
- encapsulation
- abstraction
- inheritance
- polymorphism

Learning outcome 3

The learner will:

3 Be able to debug and write code

The learner can:

3.1 Identify how to write and debug interpreted code
3.2 Identify how to write and debug compiled code
3.3 Identify issues that can occur cross-platform
3.4 Implement a solution in:
- interpreted code
- compiled code

3.5 Distinguish the differences between the coding solutions

Key word(s)

Examples of **languages:** C#, Java.

Assessment guidance

Delivery and assessment
1.2 The learner must select a fit for purpose development environment for different languages and offer adequate reasoning for the selection of the tool chosen. The learner could use IDEs (Integrated Development Environment) but is not limited to this. The following should be considered in choosing: • compatibility • productivity • extensibility • performance. 1.3 The learner should describe what is meant by source control and show knowledge of the reasons why it is important within software development. They should be able to describe the key common features of most Source Control Management (SCM) solutions and the potential impact of not using one. The learner should acknowledge the existence of a SCM strategy (such as the GitFlow branching strategy). 1.4 The learner should evaluate a solution for source control. The learner should consider the following: • language/supplier • popularity • hosting (online vs on premises) • fit for purpose • tooling • ease of use • features. 2.2 Within the explanations, the learner will also include: • OOD (object-oriented design) • the concept of instantiation • loose and tight coupling. 3.1 The learner will identify how to write and debug interpreted code, considering: • how the code is written and structured • the role of an interpreter • the steps taken when the code is executed and how the code can be debugged. 3.2 The learner should be able to identify: • how the code is written and structured • the role of a compiler • the steps taken before the code can be executed • how the code can be debugged • why there is a necessity of compilation for some languages. 3.4 For interpreted code, the learner should be given a small task, such as squaring 2 numbers. The learner should be able to write code in an interpreted language that is functional. For compiled code,

the learner should be given the same small problem and should be able to create a similar solution (an object-oriented approach is preferred). 3.5 The learner should be able to distinguish the differences between the 2 solutions and understand some of the advantages/disadvantages of compilation against interpretation. The Explanation of terms (Section 3) explains how the terms used in the unit content are applied to this qualification.
Types of evidence
Evidence could include: • research • reports • learner written statement • written or oral questions and answers • discussion • assignment • presentation.

Unit 04 Software testing (T/618/0980)

Unit summary	In this unit, the learner will understand different levels and methods of testing and what is meant by test driven development. The learner will also be able to identify and fix bugs using appropriate techniques and reflect on how the bug could have been prevented.
Guided learning hours	40
Level	3
Mandatory/optional	Mandatory

Learning outcome 1

The learner will:

1 Understand different levels, types and methods of testing

The learner can:

1.1 Identify different levels, types and methods of testing
1.2 Explain the types and methods of testing used within the stages of the Software Development Lifecycle
1.3 Explain the differences between functional and non-functional testing

Learning outcome 2

The learner will:

2 Know about test driven development

The learner can:

2.1 Explain what is meant by test driven development
2.2 Describe the structural composition of a unit test
2.3 Explain the advantages and disadvantages of unit testing
2.4 Consider what would make a bad unit test

Learning outcome 3

The learner will:

3 Be able to identify and fix a bug

The learner can:

3.1 Explain techniques which can be used to identify and fix bugs
3.2 Describe the benefits of the following techniques:
- variable watching
- compiler warnings
- unit test

3.3 Diagnose the cause of a bug using the following techniques:
- variable watching
- compiler warnings
- unit test

3.4 Perform a fix on broken code
3.5 Reflect on how the bug could have been prevented

Assessment guidance

Delivery and assessment
1.1 The learner should be able to identify the generally known levels, types and methods of testing. 1.2 The learner should be able to explain the relationship between different levels, types and methods of testing, including black box and white box methods, where and why they are used and who they are executed by. The learner should include an explanation of a minimum of 4 methods from examples of: • static testing • unit testing • integration testing • system testing • sanity testing • smoke testing • interface testing • regression testing • beta/acceptance testing. The above list is not exhaustive. 1.3 The learner should be able to explain 4 differences from functional and non-functional testing, such as: • Non-functional: - load testing - performance testing - usability testing - security testing - volume testing. • Functional: - unit testing - integration testing - system testing - sanity testing - smoke testing - interface testing - regression testing - beta/acceptance testing. 2.1 The learner should be able to offer an explanation of what is meant by TDD (test-driven development), and should understand the steps required in taking a TDD approach. 2.2 The learner should describe the structural composition of a unit test, and must include what is meant by: • setup • arrange • act

- assert
- teardown.

2.3 The learner should explain the advantages and disadvantages that a TDD approach can bring.

2.4 The learner should know what would make a bad unit test, considering the following:

- tests with complicated setup and dependencies
- testing multiple things within a single test
- use of specific dates or details within tests
- tests that can pass, without the feature working.

3.1 The learner should explain techniques that can be used to identify and fix bugs. This should include:

- steps taken to ensure the bug is a valid bug
- steps taken to find the source of the problem
- steps taken to fix the problem
- steps taken to ensure the fix is correct
- steps taken to ensure similar problems will not arise in the future.

3.2 The learner must be able to describe variable watching, compiler warnings and unit tests, explaining:

- what each technique is and how it is performed
- when each technique might be helpful
- when each technique might not be helpful.

3.4 The learner should be able to fix a problem and ensure the code runs successfully as per expectations.

3.5 The learner should be able to include the problem, a fix for the problem and an explanation as to how the problem occurred, reflecting on how it may have been prevented and what steps can be made to try and prevent similar problems occurring in the future.

The Explanation of terms (Section 3) explains how the terms used in the unit content are applied to this qualification.

Types of evidence

Evidence could include:

- research
- reports
- learner written statement
- written or oral questions and answers
- discussion
- assignment
- presentation
- sample code.

Unit 05 Understand deployment, maintenance and configuration management (A/618/0981)

Unit summary	In this unit, the learner will understand about deployment and maintenance in coding and the importance of deployment planning. They will also know about configuration management and version control systems.
Guided learning hours	30
Level	3
Mandatory/optional	Mandatory

Learning outcome 1

The learner will:

1 Know about deployment

The learner can:

1.1 Explain how to package an artefact
1.2 Explain the importance of deployment planning
1.3 Explain the difference between build and deploy
1.4 Identify the advantages and disadvantages of canary deployment
1.5 Describe what is meant by deployment strategy
1.6 Describe what is meant by continuous deployment
1.7 Identify what is meant by rollback strategy

Learning outcome 2

The learner will:

2 Understand maintenance in coding

The learner can:

2.1 Identify the key elements of maintenance
2.2 Explain why documentation of maintenance is important
2.3 Explain the improvements that could be made in maintenance
2.4 Explain why change management is used in maintenance

Learning outcome 3

The learner will:

3 Understand configuration management and version control systems

The learner can:

3.1 Explain what is meant by configuration management
3.2 Explain the tools and techniques used in configuration management systems
3.3 Explain the main features and benefits of version control systems

Assessment guidance

Delivery and assessment
1.1 The learner should be able to explain what packages and artefacts are and the relationship between the terms. The learner should possess an understanding of the process of how a package is created and its purpose within deployment. 1.4 The learner should familiarise themselves with the term ‘canary deployment’ and should be able to identify the advantages and disadvantages this form of deployment may have. 2.1 The learner should be able to identify the generally known 4 key elements of maintenance. 2.2 The learner should be able to explain what types of documentation are important to Maintenance Engineers and what should be created by Maintenance Engineers. The learner should understand what is meant by ‘live’ documentation and what the potential benefits and pitfalls are for keeping/not keeping documentation. 2.3 The learner should be able to explain what improvements can be made by a Maintenance Engineer, under what conditions improvements should be made, and what potential negative consequences could occur as part of carrying out improvement work. 2.4 The learner must cover what change management processes might be followed whilst maintaining software and what benefits and potential problems could arise from the use of change management. 3.1 The learner should be able to provide a broad explanation of what is meant by the term ‘configuration management’. 3.2 The learner should be able to explain some of the tools and techniques used in configuration management, showing an understanding of the benefits of SCM (Supply Change Management). 3.3 The learner should be able to explain the main features and benefits of version control, showing an understanding of what issues could arise for a Software Development team of not using version control processes. The Explanation of terms (Section 3) explains how the terms used in the unit content are applied to this qualification.
Types of evidence
Evidence could include: • research • reports • learner written statement • written or oral questions and answers • discussion • assignment • presentation.

Section 3

Explanation of terms

Explanation of terms

This table explains how the terms used at Level 3 in the unit content are applied to this qualification (not all verbs are used in this qualification).

Apply	Explain how existing knowledge can be linked to new or different situations in practice.
Analyse	Break the subject down into separate parts and examine each part. Show how the main ideas are related and why they are important. Reference to current research or theory may support the analysis.
Clarify	Explain the information in a clear, concise way.
Classify	Organise according to specific criteria.
Collate	Collect and present information arranged in sequence or logical order.
Compare	Examine the subjects in detail and consider the similarities and differences.
Critically compare	This is a development of compare, where the learner considers the positive aspects and limitations of the subject.
Consider	Think carefully and write about a problem, action or decision.
Demonstrate	Show an understanding by describing, explaining or illustrating using examples.
Describe	Write about the subject giving detailed information in a logical way.
Develop (a plan/idea which…)	Expand a plan or idea by adding more detail and/or depth of information.
Diagnose	Identify the cause based on valid evidence.
Differentiate	Identify the differences between two or more things.
Discuss	Write a detailed account giving a range of views or opinions.
Distinguish	Explain the difference between two or more items, resources or pieces of information.
Draw conclusions (which…)	Make a final decision or judgement based on reasons.
Estimate	Form an approximate opinion or judgement using previous knowledge or considering other information.
Evaluate	Examine strengths and weaknesses, arguments for and against, and/or similarities and differences. Judge the evidence from the different perspectives and make a valid conclusion or reasoned judgement. Reference to current research or theory may support the evaluation.
Explain	Provide detailed information about the subject with reasons showing how or why. Responses could include examples to support these reasons.

Extrapolate	Use existing knowledge to predict possible outcomes which might be outside the norm.
Identify	Recognise and name the main points accurately (some description may also be necessary to gain higher marks when using compensatory marking).
Implement	Explain how to put an idea or plan into action.
Interpret	Explain the meaning of something.
Judge	Form an opinion or make a decision.
Justify	Give a satisfactory explanation for actions or decisions.
Plan	Think about and organise information in a logical way using an appropriate format.
Perform	Carry out a task or process to meet the requirements of the question.
Provide	Identify and give relevant and detailed information in relation to the subject.
Review and revise	Look back over the subject and make corrections or changes.
Reflect	Learners should consider their actions, experiences or learning and the implications of this for their practice and/or professional development.
Select	Make an informed choice for a specific purpose.
Show	Supply evidence to demonstrate accurate knowledge and understanding.
State	Give the main points clearly in sentences or paragraphs.
Summarise	Give the main ideas or facts in a concise way.

Section 4

Additional information

Additional information

Resource requirements

To assist in the delivery of this qualification, learners should have access to the following mandatory resources:

- a digital device able to be used for coding (eg desktop PC or laptop)
- access to a storage medium prescribed by the organisation where the learner is in employment, or access to a storage medium where a simulated activity is undertaken
- web browser software/applications
- programming software
- Internet connectivity.

There is no requirement to use any specific software/applications or programming language. Centres are able to use any programming language and any free or paid-for software/applications as long as it allows learners to meet the assessment criteria.

Support for learners

Learner's Evidence Tracking Log (LETL)

The LETL covers the mandatory units in this qualification and it can help learners keep track of their work. This document can be downloaded free of charge from the Qualifications page on the NCFE website. You do not have to use the LETL – you can devise your own evidence tracking document instead.

Support for centres

Useful websites

Centres may find the following websites helpful for information, materials and resources to assist with the delivery of this qualification:

- Computer Hope: www.computerhope.com/jargon.htm
- Institute of Coding: instituteofcoding.org
- National Cyber Security Centre (secure development and deployment guidance) www.ncsc.gov.uk/collection/developers-collection

---

Learning resources

We offer a wide range of learning resources and materials to support the delivery of our qualifications. Please check the Qualifications page on the NCFE website for more information and to see what is available for this qualification.

Contact us

NCFE
Q6
Quorum Park
Benton Lane
Newcastle upon Tyne
NE12 8BT

Tel: 0191 239 8000*
Fax: 0191 239 8001
Email: customersupport@ncfe.org.uk
Websites: www.ncfe.org.uk

Version 1.1 June 2022
Information in this qualification specification is correct at the time of publishing but may be subject to change.

NCFE is a registered charity (Registered Charity No. 1034808) and a company limited by guarantee (Company No. 2896700).

To continue to improve our levels of customer service, telephone calls may be recorded for training and quality purposes.